Chapter VIII

Cache Management for Web-Powered Databases

Dimitrios Katsaros and Yannis Manolopoulos
Aristotle University of Thessaloniki, Greece

ABSTRACT

The Web has become the primary means for information dissemination. It is ideal for publishing data residing in a variety of repositories, such as databases. In such a multi-tier system (client - Web server - underlying database), where the Web page content is dynamically derived from the database (Web-powered database), cache management is very important in making efficient distribution of the published information. Issues related to cache management are the cache admission/replacement policy, the cache coherency and the prefetching, which acts complementary to caching. The present chapter discusses the issues, which make the Web cache management radically different than the cache management in databases or operating systems. We present a taxonomy and the main algorithms proposed for cache replacement and coherence maintenance. We present three families of predictive prefetching algorithms for the Web and characterize them as Markov predictors. Finally, we give examples of how some popular commercial products deal with the issues regarding the cache management for Web-powered databases.

INTRODUCTION

In the recent years the World Wide Web or simply the Web (Berners-Lee, Caililiau, Luotnen, Nielsen & Livny, 1994) has become the primary means for information dissemination. It is a hypertext-based application and uses the HTTP protocol for file transfers. What started as a medium to serve the needs of a specific
scientific community (that of Particle Physics), has now become the most popular application running on the Internet. Today it is being used for many purposes, ranging from pure educational to entertainment and lately for conducting business. Applications such as digital libraries, video-on-demand, distance learning and virtual stores, that allow for buying cars, books, computers etc. are some of the services running on the Web. The advent of the *XML* language and its adoption from the World Wide Web Council as a standard for document exchange has enlarged many old and fueled new applications on it.

During its first years the Web consisted of static *HTML* pages stored on the file system of the connected machines. When new needs arose, such as the E-Commerce or the need to publish data residing in other systems, e.g., databases, it was realized that we could not afford in terms of storage to replicate the original data in the Web server's disk in the form of *HTML* pages. Moreover, it would make no sense to replicate data that would never be requested. So, instead of static pages, an application program should run on the Web server to receive the requests from clients, retrieve the relevant data from the source and then pack them into *HTML* or *XML* format. Even the emerged “*semistructured*” *XML* databases, which store data directly into the *XML* format, need an application program which will connect to the DBMS and retrieve the *XML* file (or fragment). Thus, a new kind of pages, dynamically generated and a new architecture were born. We have no more the traditional couple of a Web client and a Web server, but a third part is added, the application program, running on the Web server and serving data from an underlying repository, in most of the cases being a database. This scheme is sometimes referred to as *Web-powered database* and the Web site, which provides access to a large number of pages whose content is extracted from databases, is called *data intensive Web site* (Atzeni, Mecca & Merialdo, 1998; Yagoub, Florescu, Issarny & Valduriez, 2000). The typical architecture for such a scenario is depicted in Figure 1. In this scheme there are three tiers, the database back-end, the Web/application server and the Web client. In order to generate dynamic content, Web servers must execute a program (e.g., *server-side scripting mechanism*). This program (script) connects to the DBMS, executes the client query, gets the results and packs them in *HTML/XML* form in order to return them to the user. Quite a lot of server-side scripting mechanisms have been proposed in the literature (Greenspun, 1999; Malaika, 1998). An alternative to having a program that generates *HTML* is the several forms of *annotated HTML*. The *annotated HTML*, such as *PHP, Active Server Pages, Java Server Pages*, embeds scripting commands in an *HTML* document.

The popularity of the Web resulted in heavy traffic in the Internet and heavy load on Web the servers. For Web-powered databases the situation is worsened by the fact that the application program must interact with the underlying database to retrieve the data. So, the net effect of this situation is network congestion, high client perceived latency, Web server overload and slow response times for Web servers. Fortunately the situation is not incurable due to the existence of *reference locality* in Web request streams. The *principle of locality* (Denning & Schwartz, 1972)
Informational and Computational Equivalence in Comparing Information Modeling Methods
Keng Siau (2004). *Journal of Database Management* (pp. 73-86).
[www.igi-global.com/article/informational-computational-equivalence-comparing-information/3306?camid=4v1a](www.igi-global.com/article/informational-computational-equivalence-comparing-information/3306?camid=4v1a)